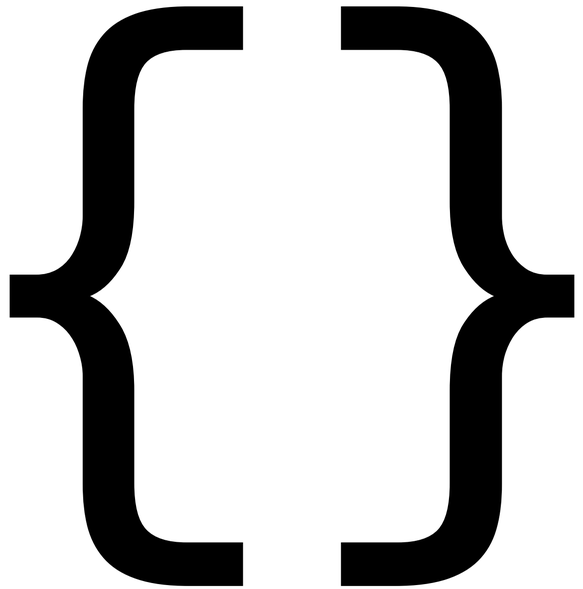
BTVJS

RUDIMENTS



```
me = {
   name:      “Nick Husher”
   twitter:   “@teslanick”
   github:    “git.io/tn”
   employer: “Dealer.com”
}
```

```
me.name      ===   “Nick Husher”
me.twitter   ===   “@teslanick”
```

```
me.name    = “Nick”
me.twitter = “@teslanick”
```

```
me          = new Object()

me.name     = “Nick”
```

Objects have relationships

```
me = {}

“” + me      // ‘[object Object]’

me.toString()
me.valueOf()
```

Object.prototype

```
me                = Object.create(null)
me.toString // undefined
me.valueOf  // undefined
"" + me     // ???
```

TypeError:

Cannot convert object
to primitive value

Object.prototype

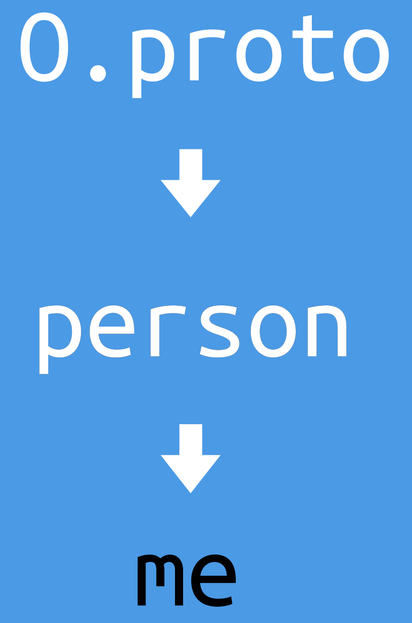
```
{
  toString: function() {},
  hasOwnProperty: function() {},
  valueOf: function() {}
}
```

{}

null

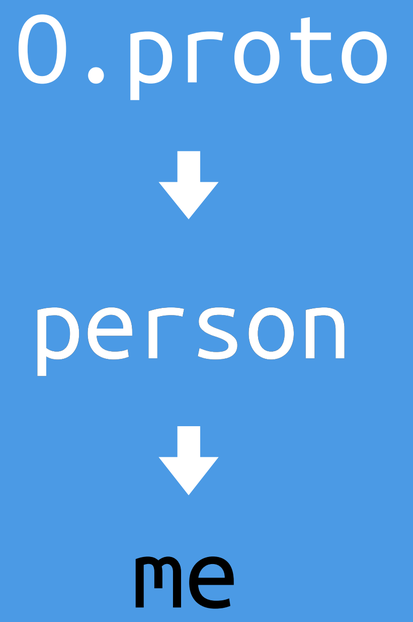
Object.create(null)

DIY prototype




```
personPrototype = {
  display: function() {
    return “My name is “ + this.name
  }
}

me          = Object.create(personPrototype)
me.name     = “Nick”
```




```
me.toString()     // “[object Object]”
me.display()      // “My name is Nick”
```

Don’t get carried away

new

Just a different

`Object.create`

```
me              = new Object()

"" + me         // "[object Object]"
```

```
function Person() {}

Person.prototype.display = function() {
  return “My name is “ + this.name
}
```

```
me                  = new Person()
me.name             = “Nick”

me.display()  // “My name is Nick”
me.toString() // “[object Object]”
```

```
function Person(name) {
    this.name = name
}
Person.prototype.display = function() {
  /*...*/
}
```

```
me                  = new Person(“Nick”)
me.display()  // “My name is Nick”
me.toString() // “[object Object]”
```

```
function Programmer() {}

Programmer.prototype =
    Object.create(Person.prototype)

Programmer.prototype.lang = function() {
    return this.language + “ rocks!”
}
```

```
me                = new Programmer()
me.name           = “Nick”
me.language       = “Javascript”

me.lang()         // “Javascript rocks!”
me.display()      // “My name is Nick”
me.toString()     // “[object Object]”
```

```
function Programmer(name, language) {
    Person.call(this, name);
    this.language = language;
}

// ...
```

```
me = new Programmer(“Nick”, “JS”)

me.lang()      // “JS rocks!”
me.display()   // “My name is Nick”
me.toString() // “[object Object]”
```

Recap

References

- Constructor chaining in Javascript: http://bit.ly/18DAm9D
- Prototypes and class-like objects: http://bit.ly/1bwrzTz
- MDN Object reference: http://mzl.la/IG0YNC